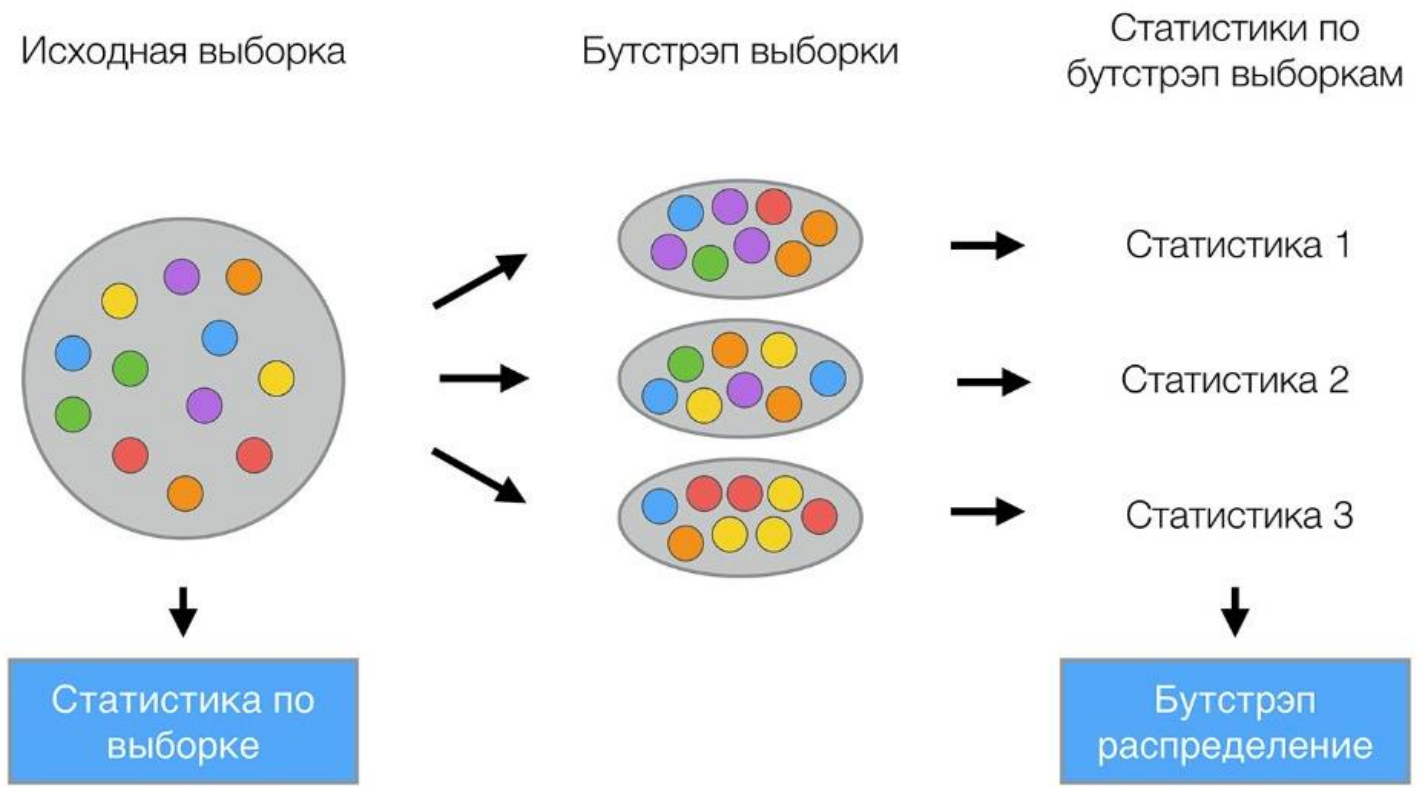
Бутстрэп

Bagging (от Bootstrap aggregation) — это один из первых и самых простых видов ансамблей. Он был придуман Лео Бре́йманом в 1994 году. Бэггинг основан на статистическом методе бутстрэппинга, который позволяет оценивать многие статистики сложных моделей.

Метод бутстрэпа заключается в следующем. Пусть имеется выборка X размера N. Равномерно возьмем из выборки N объектов с возвращением. Это означает, что мы будем N раз выбирать произвольный объект выборки (считаем, что каждый объект «достается» с одинаковой вероятностью $\frac{1}{N}$), причем каждый раз мы выбираем из всех исходных N объектов. Можно представить себе мешок, из которого достают шарики: выбранный на каком-то шаге шарик возвращается обратно в мешок, и следующий выбор опять делается равновероятно из того же числа шариков. Отметим, что из-за возвращения среди них окажутся повторы. Обозначим новую выборку через X_1. Повторяя процедуру M раз, сгенерируем M подвыборок $X_1, \ldots, X_M$. Теперь мы имеем достаточно большое число выборок и можем оценивать различные статистики исходного распределения.



Рассмотрим задачу регрессии с базовыми алгоритмами $b_1(x), \ldots, b_n(x)$. Предположим, что существует истинная функция ответа для всех объектов $y(x)$, а также задано распределение на объектах $p(x)$. В этом случае мы можем записать ошибку каждой функции регрессии

$$\varepsilon_i(x) = b_i(x) \text{Unknown character} \text{Unknown character} \text{Unknown character} y(x),\ i = 1, \ldots, n$$

и записать матожидание среднеквадратичной ошибки

$$E_x(b_i(x) \text{Unknown character} \text{Unknown character} \text{Unknown character} y(x))^2 = E_x\varepsilon_i(x).$$

Средняя ошибка построенных функций регрессии имеет вид

$$E_1 = \frac{1}{n} E_x \sum_{i=1}^{n} \varepsilon_i^2(x)$$

Предположим, что ошибки несмещены и некоррелированы:

$$\begin{aligned} E_x\varepsilon_i(x) &= 0, \\ E_x\varepsilon_i(x)\varepsilon_j(x) &= 0,\ i \neq j. \end{aligned}$$

Построим теперь новую функцию регрессии, которая будет усреднять ответы построенных нами функций:

$$a(x) = \frac{1}{n} \sum_{i=1}^{n} b_i(x)$$

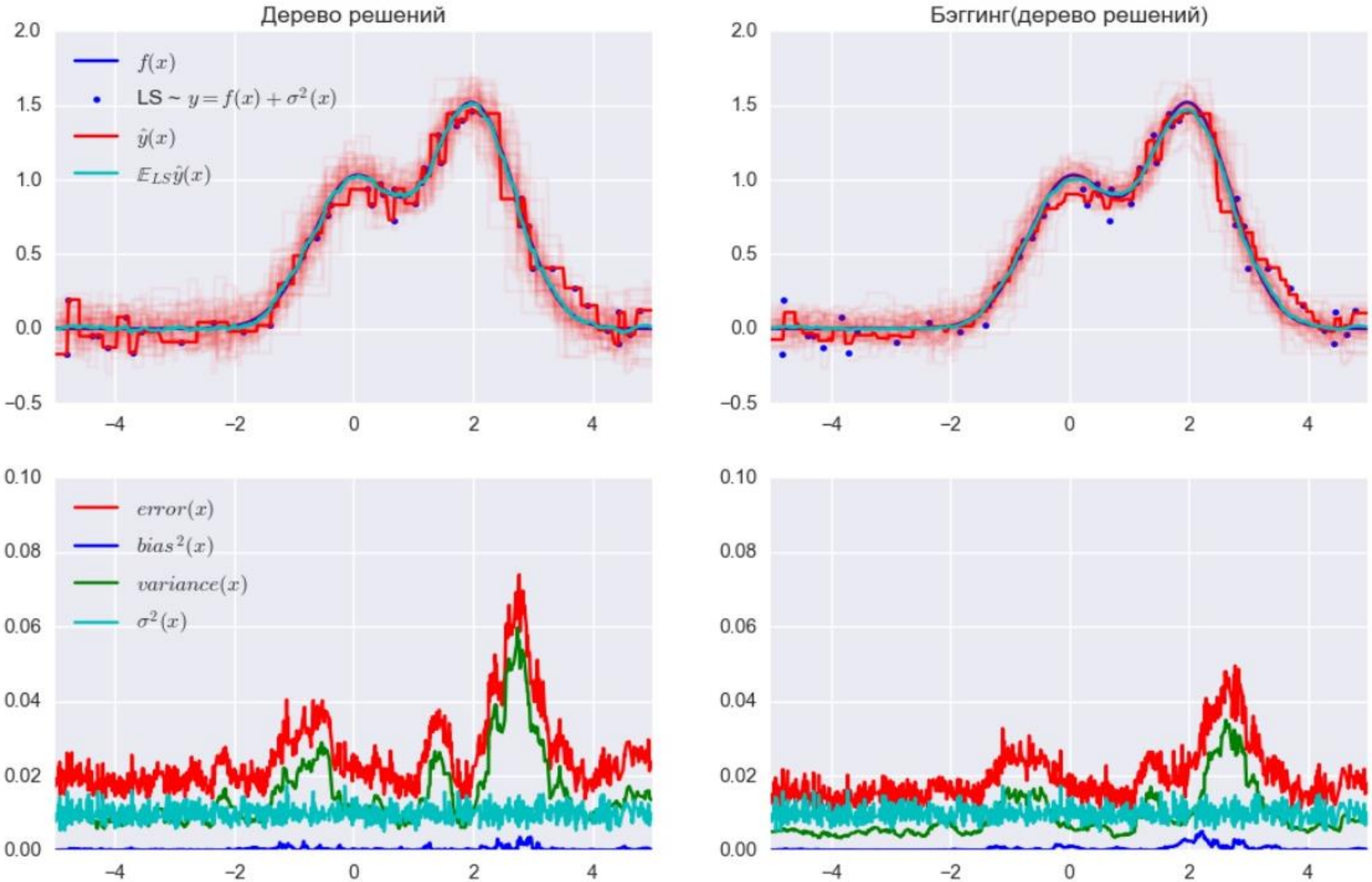
Найдем ее среднеквадратичную ошибку:

$$\begin{aligned} E_n &= E_x\Big(\frac{1}{n}\sum_{i=1}^{n} b_i(x) - y(x)\Big)^2 \\ &= E_x\Big(\frac{1}{n}\sum_{i=1}^{n} \varepsilon_i\Big)^2 \\ &= \frac{1}{n^2} E_x\Big(\sum_{i=1}^{n} \varepsilon_i^2(x) + \sum_{i \neq j} \varepsilon_i(x)\varepsilon_j(x)\Big) \\ &= \frac{1}{n} E_1 \end{aligned}$$

$$
\begin{aligned}
\mathrm{Err}\left(\vec{x}\right) &= \mathbb{E}\left[\left(y - \hat{f}\left(\vec{x}\right)\right)^2\right] \\
&= \sigma^2 + f^2 + \mathrm{Var}\left(\hat{f}\right) + \mathbb{E}\left[\hat{f}\right]^2 - 2f\,\mathbb{E}\left[\hat{f}\right] \\
&= \left(f - \mathbb{E}\left[\hat{f}\right]\right)^2 + \mathrm{Var}\left(\hat{f}\right) + \sigma^2 \\
&= \mathrm{Bias}\left(\hat{f}\right)^2 + \mathrm{Var}\left(\hat{f}\right) + \sigma^2
\end{aligned}
$$

Бэггинг позволяет снизить дисперсию (variance) обучаемого классификатора, уменьшая величину, на сколько ошибка будет отличаться, если обучать модель на разных наборах данных, или другими словами, предотвращает переобучение. Эффективность бэггинга достигается благодаря тому, что базовые алгоритмы, обученные по различным подвыборкам, получаются достаточно различными, и их ошибки взаимно компенсируются при голосовании, а также за счёт того, что объекты-выбросы могут не попадать в некоторые обучающие подвыборки.

В библиотеке `scikit-learn` есть реализация `BaggingRegressor` и `BaggingClassifier`, которая позволяет использовать большинство других алгоритмов "внутри". Рассмотрим на практике как работает бэггинг и сравним его с деревом решений,


Дерево решений
Бэггинг(дерево решений)
$f(x)$
LS ~ $y = f(x) + \sigma^2(x)$
$\hat{y}(x)$
$\mathbb{E}_{LS}\hat{y}(x)$
$error(x)$
$bias^2(x)$
$variance(x)$
$\sigma^2(x)$

Ошибка дерева решений

$$0.0255(Err) = 0.0003(Bias^2) + 0.0152(Var) + 0.0098(\sigma^2)$$

Ошибка бэггинга

$$0.0196(Err) = 0.0004(Bias^2) + 0.0092(Var) + 0.0098(\sigma^2)$$

По графику и результатам выше видно, что ошибка дисперсии намного меньше при бэггинге, как мы и доказали теоретически выше.

Бэггинг эффективен на малых выборках, когда исключение даже малой части обучающих объектов приводит к построению существенно различных базовых классификаторов. В случае больших выборок обычно генерируют подвыборки существенно меньшей длины.

Следует отметить, что рассмотренный нами пример не очень применим на практике, поскольку мы сделали предположение о некоррелированности ошибок, что редко выполняется. Если это предположение неверно, то уменьшение ошибки оказывается не таким значительным. В следующих лекциях мы рассмотрим более сложные методы объединения алгоритмов в композицию, которые позволяют добиться высокого качества в реальных задачах.

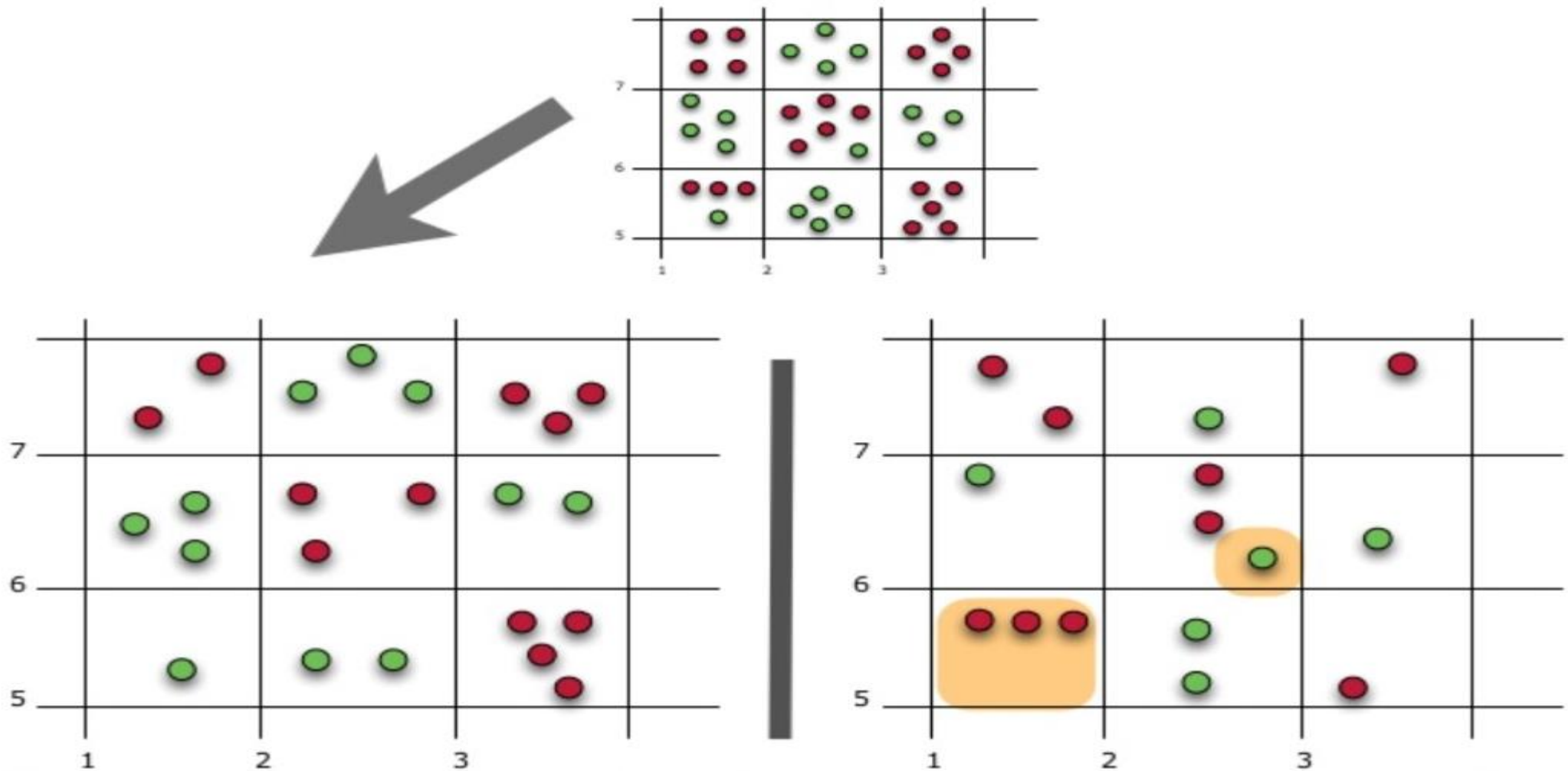
Out-of-bag error

Забегая вперед, отметим, что при использовании случайных лесов нет необходимости в кросс-валидации или в отдельном тестовом наборе, чтобы получить несмещенную оценку ошибки набора тестов. Внутренняя оценка во время работы получается следующим образом:

Каждое дерево строится с использованием разных образцов бутстрэпа из исходных данных. Примерно 37% примеров остаются вне выборки бутстрэпа и не используется при построении k-го дерева.

Это можно легко доказать: пусть в выборке ℓ объектов. На каждом шаге все объекты попадают в подвыборку с возвращением равновероятно, т.е отдельный объект — с вероятностью $\frac{1}{\ell}$. Вероятность того, что объект НЕ попадет в подвыборку (т.е. его не взяли ℓ раз): $\left(1-\frac{1}{\ell}\right)^{\ell}$. При $\ell \to +\infty$ получаем один из "замечательных" пределов $\frac{1}{e}$. Тогда вероятность попадания конкретного объекта в подвыборку $\approx 1-\frac{1}{e} \approx 63\%$.

OOBE



На рисунке видно, что наш классификатор ошибся в 4 наблюдениях, которые мы не использовали для тренировки. Значит точность нашего классификатора: $\frac{11}{15} * 100\% = 73.33\%$

Получается, что каждый базовый алгоритм обучается на ~63% исходных объектов. Значит, на оставшихся ~37% его можно сразу проверять. Out-of-Bag оценка — это усредненная оценка базовых алгоритмов на тех ~37% данных, на которых они не